Software Testing and Verification

Problem Set 6: Predicate Transforms – Solution Notes

1. a. Would. [wp(s, $\mathbf{y=z}$) = $z>-5$] => [$\{t=5 \wedge -5<z<0\}$ s $\{\mathbf{y=z}\}$ *strongly*] => [$\{t=5 \wedge z<0\}$ s $\{\mathbf{y=z+1} \wedge t=z\}$ is FALSE] since we know s will halt for at least four initial (integer) values of $z<0$ with post-condition ($\mathbf{y=z+1} \wedge t=z$) being false.

   b. Would not. [wlp(s, $\mathbf{y=z}$) = $z>-5$] => [$\{t=5 \wedge z>-5\}$ s $\{\mathbf{y=z}\}$] *weakly*. But this does **not** imply that $\{t=5 \wedge z<0\}$ s $\{\mathbf{y=z+1} \wedge t=z\}$ is FALSE since s may not terminate for $-5<z<0$.

   c. Would. In order for $\{t=5 \wedge z<0\}$ s $\{y=z+1 \wedge t=z\}$ to hold weakly, it is necessary that ($t=5 \wedge z<0$) => wlp(s, $y=z+1 \wedge t=z$). But if ($t=5 \wedge z<0$) ≠> wlp(s, $y=z+1$ $\mathbf{\vee}$ $t=z$), then it would also be the case that ($t=5 \wedge z<0$) ≠> wlp(s, $y=z+1$ $\boldsymbol{\wedge}$ $t=z$), which is an even stronger condition.

   d. Would not. ["sp(s, $t=5 \wedge z>-5$) is undefined"] implies that s does not terminate when $t=5 \wedge z>-5$ initially, which is consistent with the given assertion of ***weak*** correctness for $-5<z_0<0$. [sp(s, $t=5 \wedge z\le-5$) => ($y=z+1 \wedge t=z$)] implies that the specified post-condition, ($y=z+1 \wedge t=z$), will hold on termination for $z_0\le-5$, which is *also* consistent with the given assertion.

2. For program S:

```
           temp := x
S1         x := y
           y := temp
           if temp>z then
   S2.1       y := z
              z := temp
S2            if x>y then
                 temp := x
   S2.2          x := y          S2.2.1
                 y := temp
              end_if
           end_if
```

   we need to determine:

   wp(S, $x\le y<z$) = wp(S1; S2, $x\le y<z$)

   = wp(S1, wp(S2, $x\le y<z$))

   = wp(S1, wp(if temp>z then S2.1;S2.2, $x\le y<z$))

   = wp(S1, (temp>z $\wedge$ wp(S2.1;S2.2, $x\le y<z$)) $\vee$
   (temp$\le$z $\wedge$ $x\le y<z$))

$= wp(S1, (temp>z \wedge wp(S2.1, wp(S2.2, x\le y<z))) \vee$
$\quad (temp\le z \wedge x\le y<z))$
$= wp(S1, (temp>z \wedge wp(S2.1, wp(\text{if } x>y \text{ then } S2.2.1, x\le y<z))) \vee$
$\quad (temp\le z \wedge x\le y<z))$
$= wp(S1, (temp>z \wedge wp(S2.1, (x>y \wedge wp(S2.2.1, x\le y<z)) \vee$
$\quad\quad (x\le y \wedge x\le y<z)) \vee$
$\quad (temp\le z \wedge x\le y<z))$
$= wp(S1, (temp>z \wedge wp(S2.1, (x>y \wedge y\le x<z) \vee$
$\quad\quad (x\le y<z)) \vee$
$\quad (temp\le z \wedge x\le y<z)))$

$= wp(S1, (temp>z \wedge wp(S2.1, y<x<z \vee x\le y<z)) \vee$
$\quad (temp\le z \wedge x\le y<z))$

$= wp(S1, (temp>z \wedge (z<x<temp \vee x\le z<temp)) \vee$
$\quad (temp\le z \wedge x\le y<z))$

$= (x>z \wedge (z<y<x \vee y\le z<x)) \vee (x\le z \wedge y\le x<z)$

$= z<y<x \vee y\le z<x \vee y\le x<z$

3. ({P} s {Q}) does not imply (P=>wp(s,Q)) because weak correctness (which does NOT require that s terminate) does not imply strong correctness (which DOES require that s terminate).

4. a. wlp rule for while-do statements:

wlp(while b do S, Q) ≡ wp(while b do S, Q) V ¬wp(while b do S, true)

i. determining wp(while b do S, Q):

H_0: $J=Y \wedge Z=XY$

H_1: $J<>Y \wedge wp(s, J=Y \wedge Z=XY)$
$\quad = J=Y-1 \wedge Z=X(Y-1)$

H_2: $J<>y \wedge wp(s, J = Y-1 \wedge Z=X(Y-1))$
$\quad = J=Y-2 \wedge Z=X(Y-2)$

H_k: $J=Y-k \wedge Z=X(Y-k)$

Therefore, $H_0 \vee H_1 \vee H_2 \vee \ldots \vee H_k \vee \ldots$ simplifies to:

$(J=Y \wedge Z=XY) \vee (J<Y \wedge Z=XJ)$ = $\mathbf{(J\le Y \wedge Z=XJ)}$

ii. determining wp(while b do S, true):

H_0: J=Y $\wedge$ true

H_1: J<>Y $\wedge$ wp(s, J=Y)
= J=Y-1

H_2: J<>y $\wedge$ wp(s, J = Y-1)
= J=Y-2

H_k: J=Y-k

Therefore, $H_0 \vee H_1 \vee H_2 \vee \ldots \vee H_k \vee \ldots$ simplifies to: **J≤Y**

Thus, wlp(while b do S, Q) $\equiv$ (J≤Y $\wedge$ Z=XJ) V J>Y = **Z=XJ V J>Y**

b. Obviously, the wlp is weaker than the given invariant, i.e.,

(Z=XJ) => (Z=XJ V J>Y)

This makes sense since the wlp is the **weakest** condition on the initial state of program S ensuring state Q on termination *if S terminates.* If J>Y initially, the program will obviously not terminate, implying weak correctness ***whether Z=XJ holds initially or not.***

5. We wish to prove: (wlp(while b do s, Q) $\wedge$ ~b) => Q. The left hand side of the implication, (wp(while b do s, Q) $\wedge$ ~b), is:

$[\ (H_0 \vee H_1 \vee \ldots)_{\text{wlp(while b do s, } \mathbf{Q})}\ \vee \neg(H_0 \vee H_1 \vee \ldots)_{\text{wp(while b do S, } \mathbf{true})}\ ] \wedge \sim b$

$= [\ [(\sim b \wedge Q) \vee (b \wedge wp(s, H_0)) \vee (b \wedge wp(s, H_1)) \vee \ldots] \wedge \sim b] \vee$
$\neg[(\sim b \wedge true) \vee (b \wedge wp(s, \sim b)) \vee (b \wedge wp(s, H_1)) \vee \ldots] \wedge \sim b]\ ]$

$= (\sim b \wedge Q) \vee [\neg(\sim b) \wedge \neg(b \wedge wp(s, \sim b)) \wedge \neg(b \wedge wp(s, H_1)) \wedge \ldots] \wedge \sim b$

$= (\sim b \wedge Q) \vee [b \wedge (\neg b \vee \neg wp(s, \sim b)) \wedge (\neg b \vee \neg wp(s, H_1)) \wedge \ldots] \wedge \sim b$

$= (\sim b \wedge Q) \vee [\mathbf{(b \wedge \sim b)} \wedge [(\neg b \vee \neg wp(s, \sim b)) \wedge (\neg b \vee \neg wp(s, H_1)) \wedge \ldots] \wedge \sim b]$

$= (\sim b \wedge Q) \vee [\mathbf{(false)} \wedge [(\neg b \vee \neg wp(s, \sim b)) \wedge (\neg b \vee \neg wp(s, H_1)) \wedge \ldots] \wedge \sim b]$

$= (\sim b \wedge Q) \Rightarrow Q\ \surd$

6. a. $H_1 = wp(s, c \wedge Q)$
$H_2 = wp(s, \sim c \wedge H_1)$
$H_3 = wp(s, \sim c \wedge H_2)$
$H_k = wp(s, \sim c \wedge H_{k-1})$

b. $H_1 = wp(s, c \wedge Q) = wp(s, y=0 \wedge x=17)$
$= y-1=0 \wedge x+1=17$
$= y=1 \wedge x=16$

$H_2 = wp(s, \sim c \wedge H_1) = wp(s, y<>0 \wedge y=1 \wedge x=16)$
$= y=2 \wedge x=15$

$H_3 = y=3 \wedge x=14$

$H_k = y=k \wedge x=(17-k)$

wp (in closed form) = $\mathbf{y>0 \wedge x=17-y}$